Les interfaces

L'objectif

L'objectif des interfaces est de décrire un ensemble de fonctionnalités liées pouvant appartenir à n'importe quelle classe.

C'est un « contrat » qu'on peut demander à une classe de respecter.

Le langage python ne supportant nativement pas les interfaces.
Il est possible d'utiliser 2 solutions à la place :

- Le Duck Typing
- Simuler leurs fonctionnements

Le Duck Typing

Le nom de ce système de typage est une référence au test du canard :

> *« Si je vois un oiseau qui vole comme un canard, cancane comme un canard, et nage comme un canard, alors j'appelle cet oiseau un canard »*

Concrètement : si un objet défini les méthodes d'un type, c'est qu'il est de se type.

Exemple :
Un objet est une « collection » s'il défini les méthodes : __len__, __contains__, __iter__.

Simuler les interfaces

En python, il est possible de simuler des interfaces à l'aide de plusieurs mécanismes :

- Les classes abstraites
- L'héritage multiple
- Le polymorphisme

Simuler les interfaces

Pour mettre en place une *interface* en python, il faut :

- Créer une "*interface*" à l'aide d'une classe abstraite, celle-ci devra uniquement posséder les signatures des méthodes.
- Mettre en place l'héritage entre "*l'interface*" et notre classe.
- Implémenter les méthodes de "*l'interface*" au sein de notre classe.

Grâce au polymorphisme, il est possible de vérifier si un objet implémente une interface à l'aide de la méthode « isinstance(mon_objet, mon_interface) »

Merci pour votre attention.

CogniTIC

Building Together Better IT Solutions